

BAB II

TINJAUAN PUSTAKA

Penelitian yang akan dilakukan memerlukan tinjauan pustaka sebagai bahan acuan dari hasil penelitian – penelitian yang telah dilakukan sebelumnya, hal ini sangat penting agar terhindar dari plagiasi atau duplikasi dari penelitian yang sudah ada yang bertujuan sebagai bahan agar penelitian mengenai topik ini dapat terus berkembang. Berikut beberapa pemaparan tentang penelitian sebelumnya yang sudah pernah dilakukan berkaitan dengan data dan metode yang dilakukan.

2.1 Studi Terkait

Pemeliharaan yang efektif dalam industri pengolahan kelapa sawit sangat penting untuk memastikan efisiensi operasional dan keandalan produksi. Faisal, A. [9]. *Pengembangan Model Diagnosis untuk Deteksi Kegagalan Mesin Menggunakan Algoritma Random Forest dalam Klasifikasi Pembelajaran Mesin Berdasarkan Uji Non-Destructive Vibrasi*. Penelitian ini menggunakan algoritma Random Forest untuk mendeteksi kegagalan pada mesin berdasarkan data getaran. Hasil menunjukkan akurasi di atas 95%, menjadikannya alat yang andal untuk diagnosis sistem bearing. Moh. Abib Safaqqillah [1] di Universitas Hasanuddin membandingkan metode *Decision Tree Regression* dan *Random Forest Regression* untuk prediksi degradasi daya pada panel surya akibat debu. Hasil penelitian menunjukkan bahwa model Random Forest memberikan hasil yang lebih baik dalam kondisi lingkungan minim debu dengan nilai Mean Squared Error (MSE) lebih rendah dibandingkan Decision Tree. Adapun Penelitian lain di Institut

Teknologi Bandung [2] menunjukkan penerapan Random Forest untuk meningkatkan manajemen kesehatan aset di pembangkit listrik. Model ini mencapai akurasi prediksi sebesar 95% dalam memprediksi kondisi aset, dengan fitur penting seperti Maintenance Priority Index (MPI) yang berpengaruh signifikan terhadap prediksi. Formanto Paliling [3] dalam penelitiannya menjelaskan langkah-langkah penggunaan Random Forest untuk pemeliharaan prediktif mesin, termasuk pengumpulan data sensor, pra-pemrosesan data, pemilihan fitur, dan pelatihan model. Metode ini menunjukkan potensi besar dalam meningkatkan efisiensi operasional dan mengurangi biaya melalui prediksi yang akurat dengan akurasi sebesar 90%. Penggunaan algoritma Random Forest juga terbukti efektif dalam klasifikasi berbasis indikator kinerja efisiensi energi, yang dapat membantu dalam diagnosis kegagalan peralatan. Model ini dapat memberikan informasi tentang fitur-fitur yang mengalami anomali, sehingga memprioritaskan pekerjaan pemeliharaan dengan akurasi sebesar 92.60%. Rosa Vella Erdizon [4] mengkaji berbagai metode pemeliharaan, termasuk *Failure Mode and Effects Analysis (FMEA)*, *Reliability-Centered Maintenance (RCM)*, dan *Overall Equipment Effectiveness (OEE)*. Penelitian ini mengidentifikasi tren perawatan mesin dari artikel yang dipublikasikan antara 2010 hingga 2024, menunjukkan bahwa pendekatan komprehensif dalam pemeliharaan dapat meningkatkan efisiensi operasional dan keandalan produksi. Dengan menerapkan metode ini, pabrik dapat mengantisipasi dan mengurangi risiko kegagalan mesin, yang pada gilirannya berkontribusi pada peningkatan produktivitas.

Dalam konteks yang lebih spesifik, Dicky Dwi Armanda et al. [5] menerapkan RCM di PT. XYZ untuk mengevaluasi komponen mesin dan penjadwalan perbaikan. Hasil penelitian menunjukkan bahwa pemeliharaan yang baik dapat meningkatkan produktivitas dengan mengidentifikasi dan memperbaiki masalah mesin secara terjadwal. Penelitian ini menegaskan pentingnya strategi pemeliharaan yang terencana untuk meminimalkan downtime dan memastikan kelancaran operasional.

Selain itu, Essa Abubakar Wahid et al. [6] mengembangkan algoritma *Predictive Maintenance* menggunakan *Machine Learning* untuk *Coal Pfister Feeder*. Penelitian ini bertujuan untuk memprediksi waktu optimal untuk perawatan sebelum kegagalan terjadi. Hasil evaluasi menunjukkan bahwa algoritma Gradient Boosting memiliki akurasi terbaik dibandingkan model lainnya, menyoroti potensi teknologi machine learning dalam meningkatkan efektivitas pemeliharaan prediktif.

Dwi Kusumaningrum [7] juga melakukan penelitian tentang penggunaan *Machine Learning* untuk *Predictive Maintenance* di PT X, dengan fokus pada model prediksi diagnosis dan prognosis. Hasilnya menunjukkan bahwa algoritma *Random Forest* dengan parameter optimal memiliki performa lebih baik dibandingkan dengan *Support Vector Machine (SVM)* dan *Artificial Neural Network (ANN)*. Penelitian ini menunjukkan bahwa pemanfaatan algoritma machine learning dapat memberikan wawasan yang lebih baik dalam proses pemeliharaan.

Selanjutnya, Fajar Martha Subqi dan Dyah Anggraini [8] menggunakan metode *Naïve Bayes Classifier* untuk pemeliharaan prediktif mesin produksi berdasarkan data kerusakan. Dengan menggunakan data dari sepuluh mesin yang sering mengalami kerusakan, penelitian ini berhasil mencapai akurasi 93,65% dalam menentukan mesin yang memerlukan kegiatan *preventive maintenance*. Hasil ini menunjukkan bahwa pendekatan berbasis data dapat secara signifikan meningkatkan efektivitas pemeliharaan. Secara keseluruhan, penelitian-penelitian ini menunjukkan bahwa penerapan metode pemeliharaan yang tepat, baik itu melalui pendekatan tradisional seperti FMEA dan RCM maupun teknologi modern seperti machine learning, dapat meningkatkan efisiensi operasional dan keandalan produksi di berbagai industri. Pendekatan komprehensif dalam pemeliharaan tidak hanya mengurangi risiko kegagalan tetapi juga meningkatkan produktivitas dan keberlanjutan operasional.

2.2 Mesin Eye Forming Robot

Leaf spring adalah komponen suspensi yang terdiri dari beberapa lembar baja yang disusun secara bertumpuk. Komponen ini berfungsi untuk menyerap guncangan dan memberikan stabilitas pada kendaraan. Bagian "eye" pada *leaf spring* adalah lubang yang dibentuk di kedua ujungnya, yang berfungsi sebagai titik penghubung ke sasis kendaraan. *Eye Forming Robot* adalah mesin otomatis yang digunakan dalam industri manufaktur pegas untuk membentuk mata (*eye*) pada ujung pegas daun. Mesin ini merupakan bagian integral dari proses produksi *leaf spring* di PT Indospring Tbk. Menggunakan teknologi robotik dan sistem kontrol otomatis untuk memastikan presisi dan hasil konsisten. Penggunaan teknologi

robotik dalam mesin *eye forming* memberikan beberapa keuntungan diantaranya, proses yang sebelumnya dilakukan secara manual kini dapat dilakukan secara otomatis, mengurangi waktu produksi dan meningkatkan efisiensi, robot dapat melakukan tugas dengan tingkat akurasi yang sangat tinggi, mengurangi variasi dalam produk akhir dan memastikan bahwa setiap leaf spring memenuhi standar kualitas yang ketat.

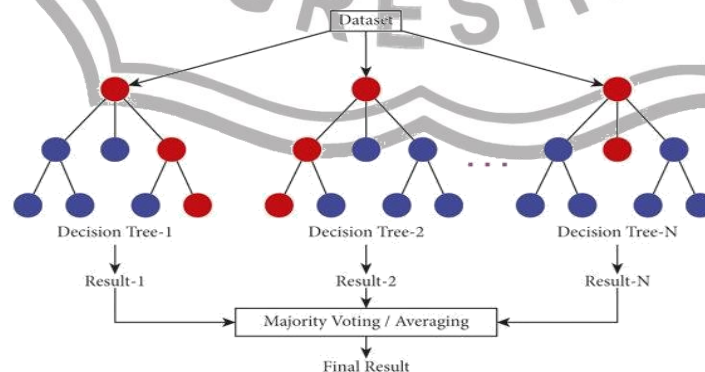
2.3 Predictive Maintenance

Predictive maintenance merupakan metode pemeliharaan yang memanfaatkan analisis data dan teknologi canggih untuk memprediksi potensi kerusakan atau kegagalan pada mesin atau peralatan. Dalam metode ini, sensor-sensor dipasang pada peralatan yang dipantau untuk mengumpulkan data secara *real-time*. Data tersebut kemudian dianalisis menggunakan algoritma dan teknik *machine learning* guna mendeteksi pola, tren, atau anomali yang dapat menunjukkan kemungkinan kerusakan di masa depan. Dengan teknologi ini, perusahaan dapat melakukan tindakan *preventif* sebelum kegagalan terjadi, seperti melakukan perbaikan atau pemeliharaan terjadwal. Hal ini membantu mencegah kerugian akibat *downtime* yang tidak terduga, meningkatkan efisiensi operasional, dan memperpanjang umur peralatan. Selain itu, *predictive maintenance* juga memungkinkan perusahaan untuk merencanakan secara lebih efektif, mengoptimalkan sumber daya, dan menekan biaya pemeliharaan secara keseluruhan. Oleh karena itu, *predictive maintenance* menjadi solusi penting dalam industri, karena dapat meningkatkan keandalan dan kinerja peralatan sekaligus mengurangi risiko kerusakan yang berpotensi merugikan perusahaan. *Machine*

learning adalah salah satu cabang kecerdasan buatan yang berfokus pada pembelajaran dari proses komputasi tanpa memerlukan pemrograman spesifik untuk menjalankan suatu tugas. Dengan kemampuannya, *machine learning* menawarkan pendekatan prediktif yang efektif untuk mendukung implementasi *Predictive Maintenance*.

2.4 Metode Random Forest

Random Forest adalah algoritma machine learning yang memanfaatkan konsep *ensemble learning* dengan membangun sejumlah besar pohon keputusan (*decision trees*) yang saling independen. *Random Forest* diperkenalkan oleh Leo Breiman dan Adele Cutler pada tahun 2001, *Random Forest* telah menjadi salah satu metode yang paling populer dan efektif untuk klasifikasi dan regresi. Setiap pohon dalam forest ini digunakan untuk melakukan prediksi, dan hasil prediksi dari semua pohon digabungkan untuk mendapatkan hasil akhir. *Ensemble learning* ini mengatasi beberapa masalah yang sering terjadi dalam pemodelan dengan satu pohon keputusan, seperti *overfitting* (memprediksi data pelatihan dengan sangat baik tetapi tidak umum), kepekaan terhadap perubahan kecil dalam data pelatihan, dan bias yang mungkin muncul dari satu pohon.



Gambar 2.1 Visualisasi Random Forest

Berdasarkan gambar yang telah disajikan di atas, Langkah dalam algoritma *random forest* adalah sebagai berikut:

1. Algoritma akan memilih sampel acak dari dataset yang disediakan.
2. Membuat *decision tree* dengan melakukan *bootstrap* sampling untuk setiap sampel yang dipilih.
3. Kemudian setiap *decision tree* akan menghasilkan prediksi menggunakan predictor, Setiap hasil prediksi akan dilakukan proses voting menggunakan nilai yang paling sering muncul.
4. Algoritma akan memilih hasil prediksi yang paling banyak dipilih sebagai prediksi akhir.

Dalam menghasilkan hasil prediksi yang akurat, random forest bekerja dengan menerapkan metode bagging (agregasi bootstrap) yang merupakan pengumpulan dari beberapa meta algoritma untuk meningkatkan tingkat akurasi dari algoritma *data mining*. Metode tersebut mengambil sampel data acak dari dataset. Sampel yang diambil dari data asli dilakukan melalui proses raw sampling. Sampel yang didapatkan akan dilakukan penggantian yang disebut dengan *bootstrap*. *Bootstrap* pada *random forest* digunakan untuk mengurangi pohon keputusan yang bias dan memutuskan pemisahan di setiap pohon keputusan.

Random Forest beroperasi berdasarkan dua prinsip utama:

- ***Bootstrap Aggregating (Bagging)***: Teknik ini melibatkan pengambilan sampel acak dari dataset pelatihan dengan pengembalian (*sampling with replacement*). Setiap pohon keputusan dibangun menggunakan subset data

yang berbeda, yang membantu mengurangi varians dan meningkatkan generalisasi model.

- **Random Feature Selection:** Saat membangun setiap pohon, Random Forest hanya mempertimbangkan subset acak dari fitur yang tersedia untuk menentukan split pada setiap node. Ini mengurangi korelasi antara pohon-pohon yang dihasilkan, sehingga meningkatkan keanekaragaman model.

Algoritma *random forest* memiliki kemampuan untuk secara acak menerima data pelatihan dari himpunan bagian dan membentuk pohon dengan algoritma acak. Penggabungan klasifikasi dari beberapa pohon perlu dilakukan pelatihan secara terpisah. Pembentukan pohon keputusan pada *random forest* dimulai dengan melakukan perhitungan nilai *entropy* yang digunakan untuk mengukur tingkat ketidakmurnian atribut dan nilai *information gain* untuk mencari variabel yang akan dipilih untuk menumbuhkan pohon (Nurfadilla & Faisal, 2022; Sandag, 2020). Adapun rumus dari nilai *entropy* ada pada persamaan 1 dan nilai *information gain* pada persamaan 2 sebagai berikut:

$$H(S) = - \sum_{i=1}^c p_i \log_2(p_i) \quad (2.1)$$

di mana:

- $H(S)$ adalah entropi dari set S ,
- p_i adalah proporsi kelas ke- i dalam set S ,
- c adalah jumlah kelas.

- *Voting/Averaging*: Setelah semua pohon dibangun, hasil prediksi dari setiap pohon digabungkan.

$$\hat{y} = \text{mode}(y_1, y_2, \dots, y_B) \quad (2.2)$$

di mana:

- \hat{y} adalah prediksi akhir,
- y_b adalah prediksi dari pohon ke- b ,
- B adalah jumlah total pohon.

Untuk klasifikasi, metode voting mayoritas digunakan, sedangkan untuk regresi, rata-rata dari semua prediksi pohon digunakan.

$$\hat{y} = \frac{1}{B} \sum_{b=1}^B y_b \quad (2.3)$$

Untuk menggambarkan proses pembangunan model *random forest*, dapat menggunakan *pseudocode* sebagai cara deskriptif dalam memberikan pandangan umum mengenai langkah-langkah yang akan diambil dalam penyelesaian suatu masalah. Adapun *pseudocode* sederhana untuk algoritma *random forest* sebagai berikut:

1. Fungsi Bangun Pohon(Random Samples): Fungsi ini membangun satu pohon keputusan berdasarkan sampel yang diberikan. Ia memeriksa kondisi berhenti, memilih subset fitur, menentukan fitur terbaik, membagi sampel, dan membangun simpul pohon.

2. Fungsi Bangun Random Forest: Fungsi ini membangun koleksi pohon keputusan. Ia mengambil sampel acak dari data dan membangun pohon untuk setiap iterasi, menyimpannya dalam koleksi pohon.
3. Fungsi PrediksiRandomForest: Fungsi ini melakukan prediksi dengan menghitung hasil dari setiap pohon dalam koleksi dan mengembalikan label mayoritas dari hasil tersebut.

Pseudocode tersebut menggambarkan proses pembangunan *random forest* dengan membangun pohon-pohon keputusan dengan menggunakan sampel acak dan fitur-fitur acak. Saat melakukan prediksi, hasil prediksi dari pohon-pohon akan diambil dan dilakukan *majority voting* sebagai *output* akhir dari *random forest*.

2.4.1 Keunggulan dan Kekurangan Random Forest

Keunggulan *Random Forest*

Random Forest memiliki beberapa keunggulan yang membuatnya populer di kalangan praktisi data:

- Akurasi Tinggi: *Random Forest* sering kali memberikan akurasi yang lebih baik dibandingkan dengan model tunggal, terutama pada dataset yang besar dan kompleks.
- Robust terhadap *Overfitting*: Meskipun setiap pohon keputusan dapat *overfit*, penggabungan banyak pohon membantu mengurangi risiko ini.
- Kemampuan Menangani Data Hilang: *Random Forest* dapat menangani data yang hilang dengan baik, karena pohon keputusan dapat dibangun meskipun beberapa nilai hilang.

- *Feature Importance*: *Random Forest* dapat memberikan informasi tentang pentingnya fitur, yang membantu dalam pemilihan fitur dan interpretasi model.

Kekurangan Random Forest

Meskipun memiliki banyak keunggulan, *Random Forest* juga memiliki beberapa kekurangan:

- **Kompleksitas Model**: *Random Forest* adalah model yang kompleks dan sulit untuk diinterpretasikan dibandingkan dengan pohon keputusan tunggal.
- **Waktu Komputasi**: Membangun banyak pohon keputusan dapat memakan waktu, terutama untuk dataset yang sangat besar.
- **Memori**: *Random Forest* memerlukan lebih banyak memori dibandingkan dengan model yang lebih sederhana, karena menyimpan banyak pohon keputusan.

2.4.2 Matrik Performa

Dalam melakukan evaluasi performa dalam pemodelan machine learning, dibutuhkan suatu ukuran yang digunakan dalam mengukur kinerja model. Fungsi dari metrik performa adalah membantu menggambarkan kinerja model. Pengukuran tersebut berdasarkan pada informasi yang diberikan oleh confusion matrix. Confusion Matrix adalah model yang memiliki nilai fungsi untuk melakukan analisis apakah klasifikasi pada metode yang dinilai memiliki label baik atau buruk (Fadiyah et al., 2022). Confusion Matrix digunakan untuk

merepresentasikan hasil dari akurasi dan merangkum performa model yang sudah dibuat.

Pengukuran kinerja model berdasarkan informasi yang diberikan oleh *confusion matrix* disebut dengan *performance metrics* (metrik performa). Beberapa metrik performa yang digunakan adalah sebagai berikut:

1. *Accuracy*

Merupakan metrik yang menggambarkan seberapa akurat model dapat mengklasifikasikan dengan benar. Dalam hal ini merupakan rasio prediksi benar dengan keseluruhan data atau tingkat kedekatan nilai prediksi dengan nilai aktual. Adapun persamaan untuk memperoleh nilai *accuracy* sebagai berikut:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.4)$$

Atau

$$Akurasi = \frac{\sum \text{True Positives}}{\text{Total Samples}} \quad (2.5)$$

2. *Precision*

Merupakan metrik yang menggambarkan tingkat keakuratan antara data yang diminta dengan hasil prediksi yang diberikan oleh model klasifikasi, dengan kata lain *precision* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif. Adapun persamaan untuk memperoleh nilai *precision* sebagai berikut:

$$precision = \frac{TP}{TP + FP} \quad (2.6)$$

3. Recall

Merupakan metrik yang menggambarkan keberhasilan dari model klasifikasi dalam menemukan informasi, dalam pengertian lain bahwa *recall* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. Adapun persamaan untuk memperoleh nilai *recall* sebagai berikut:

$$recall = \frac{TP}{TP + FN} \quad (2.7)$$

4. F1- Score

Merupakan metrik yang menggambarkan rata-rata harmonis antara nilai *precision* dengan *recall*. Semakin besar nilai F1 yang dihasilkan maka semakin baik performansinya. Adapun persamaan untuk memperoleh nilai *F1- score* sebagai berikut:

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (2.8)$$

5. Weighted Average

Weighted average mempertimbangkan jumlah sampel pada setiap kelas (support).

$$Weighted\ Average = \frac{\sum (Nilai\ Metrik \times Support)}{Total\ Support} \quad (2.9)$$

6. Macro Average

Macro average menghitung rata-rata sederhana untuk semua kelas, tanpa mempertimbangkan proporsi support.

$$\text{Macro Average} = \frac{\text{Total Nilai Metrik}}{\text{Jumlah Kelas}} \quad (2.10)$$

Keterangan :

TP = True Positive

FP = False Positive

FN = False Negative

TN = True Neagtive

2.5 Python

Python merupakan Bahasa pemrograman tingkat tinggi yang digunakan untuk pengembangan perangkat lunak, analisis data, kecerdasan buatan, pemrograman *web*, dan lain sebagainya. Karena fleksibilitas penggunaan Bahasa python, membuat Bahasa tersebut menjadi Bahasa pemrograman yang sangat populer dengan sintaksisnya yang mudah dipahami dan dibaca dan dilengkapi dengan fungsionalitas Pustaka (*library*) yang besar serta komprehensif (Syahrudin & Kurniawan, 2018). Bahasa pemrograman python dibuat oleh Guido van Rossum dan dirilis pada tahun 1991.

Bahasa pemrograman python dapat digunakan oleh pemula dalam mengotomatisasikan tugas-tugas sederhana komputer seperti mengganti nama file, mencari atau mengunduh konten serta mengirimkan email atau pesan pada waktu tertentu. Python mendukung berbagai paradigma pemrograman seperti

pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Python juga merupakan bahasa pemrograman dinamis yang memiliki manajemen memori otomatis. Meskipun sering digunakan sebagai bahasa script, Python memiliki cakupan yang lebih luas daripada sekadar skrip. Penggunaannya mencakup berbagai konteks pengembangan perangkat lunak, dan Python dapat berjalan di berbagai sistem operasi yang berbeda.

2.6 Scikit-learn

Scikit-learn atau disebut juga dengan sklearn merupakan module Bahasa pemrograman Python yang digunakan untuk membantu melakukan pemrosesan data maupun pelatihan data. Scikit-learn dibangun atas beberapa data umum atau Pustaka matematika Python seperti Numpy untuk operasi matematika berkaitan dengan matriks, SciPy untuk visualisasi data, Matplotlib untuk visualisasi data, Pandas untuk penanganan, manipulasi, dan analisis data, SymPy untuk matematika simbolik, hingga IPython sebagai konsol interaktif untuk Python. Scikit-learn juga menyediakan berbagai algoritma *machine learning* baik supervised maupun unsupervised. Pustaka tersebut dapat digunakan untuk mengimplementasikan berbagai algoritma pre-processing, *machine learning*, visualisasi, dan algoritma cross-validation. Terdapat banyak fitur yang disediakan oleh Scikit-learn seperti model klasifikasi, *clustering*, regresi, dan proses lainnya yang dapat dimanfaatkan pada tahap feature engineering. Scikit-learn juga menyediakan dokumentasi API untuk pengguna yang ingin mengintegrasikan algoritma dengan platform pengguna.

2.7 Teknologi Notifikasi WhatsApp API

WhatsApp API adalah antarmuka pemrograman aplikasi (API) yang memungkinkan pengembang untuk mengintegrasikan fungsionalitas WhatsApp ke dalam aplikasi atau sistem mereka. API ini dirancang untuk bisnis dan organisasi yang ingin berkomunikasi dengan pelanggan mereka secara efisien dan efektif. Dengan menggunakan WhatsApp API, perusahaan dapat mengirim pesan, notifikasi, dan pembaruan kepada pelanggan dalam skala besar.

Fungsi utama dari WhatsApp API meliputi:

- **Pengiriman Pesan:** Memungkinkan pengiriman pesan teks, gambar, video, dan dokumen kepada pengguna WhatsApp.
- **Notifikasi Real-Time:** Memungkinkan pengiriman notifikasi secara real-time kepada pengguna, yang sangat berguna dalam konteks sistem peringatan dini.
- **Interaksi Pelanggan:** Memfasilitasi interaksi dua arah antara perusahaan dan pelanggan, memungkinkan pelanggan untuk memberikan umpan balik atau menanyakan informasi lebih lanjut.
- **Automasi:** Memungkinkan pengembangan chatbot dan sistem otomatis yang dapat menjawab pertanyaan umum atau memberikan informasi tanpa intervensi manusia.

WhatsApp API juga mendukung pengiriman pesan yang dipersonalisasi, yang dapat meningkatkan pengalaman pengguna dan keterlibatan pelanggan. Dengan lebih dari 2 miliar pengguna aktif, WhatsApp menjadi salah satu platform komunikasi yang paling banyak digunakan di

dunia, menjadikannya pilihan yang ideal untuk bisnis yang ingin menjangkau pelanggan mereka secara langsung.

2.7.1 Implementasi WhatsApp API dalam Sistem Peringatan Dini

Implementasi WhatsApp API dalam sistem peringatan dini melibatkan integrasi API ke dalam arsitektur sistem yang dirancang untuk memantau kondisi mesin dan memberikan notifikasi kepada pengguna ketika terdeteksi potensi masalah. Dalam konteks predictive maintenance, sistem peringatan dini dapat memberikan informasi yang sangat berharga kepada teknisi dan manajer pemeliharaan untuk mengambil tindakan yang diperlukan sebelum kerusakan terjadi.

Langkah-langkah implementasi WhatsApp API dalam sistem peringatan dini meliputi:

1. **Integrasi API:** Pengembang harus mengintegrasikan WhatsApp API ke dalam sistem yang ada. Ini melibatkan pendaftaran akun bisnis di WhatsApp Business API dan mendapatkan kredensial yang diperlukan untuk mengakses API.
2. **Pengaturan Notifikasi:** Sistem harus dirancang untuk mengidentifikasi kondisi yang memerlukan perhatian, seperti data sensor yang menunjukkan anomali atau penurunan kinerja mesin. Ketika kondisi ini terdeteksi, sistem akan memicu pengiriman notifikasi.
3. **Pengiriman Pesan:** Setelah kondisi yang memerlukan perhatian terdeteksi, sistem akan menggunakan WhatsApp API untuk mengirim pesan notifikasi kepada pengguna yang relevan. Pesan ini dapat berisi informasi tentang

masalah yang terdeteksi, langkah-langkah yang harus diambil, dan kontak untuk bantuan lebih lanjut.

4. **Monitoring dan Umpan Balik:** Setelah implementasi, penting untuk memantau efektivitas sistem notifikasi. Pengguna dapat memberikan umpan balik tentang relevansi dan kecepatan notifikasi yang diterima, yang dapat digunakan untuk meningkatkan sistem di masa mendatang.
5. **Keamanan dan Privasi:** Dalam implementasi WhatsApp API, penting untuk mempertimbangkan aspek keamanan dan privasi data. Data yang dikirim melalui WhatsApp harus dienkripsi dan dilindungi untuk menjaga kerahasiaan informasi.

Dengan mengintegrasikan WhatsApp API ke dalam sistem peringatan dini, perusahaan dapat meningkatkan responsivitas dan efisiensi dalam menangani masalah yang muncul, serta memberikan informasi yang tepat waktu kepada tim pemeliharaan. Hal ini tidak hanya membantu dalam mengurangi downtime mesin tetapi juga meningkatkan produktivitas dan efisiensi operasional secara keseluruhan.